Remarks

Claims 1-16 and 17-20 are pending in the application and are presented for reconsideration without amendment. No new matter has been added.

Claim Rejections

Claims 1-5, 10-13, 15, 18 and 19 are rejected under 35 U.S.C. § 102(b) as being unpatentable over Olsen (US 2001/0032300) in view of Knothe et al. (U.S. Pat. No. 4,811,293).

Claim 6 is rejected under 35 U.S.C. § 103(a) as unpatentable over Olsen (US 2001/0032300) in view of Knothe et al. (U.S. Pat. No. 4,811,293), and further in view of Aviani Jr. (US 5,950,205).

Claim 7 is rejected under 35 U.S.C. § 103(a) as unpatentable over Olsen (US 2001/0032300) in view of Knothe et al. (U.S. Pat. No. 4,811,293), and further in view of Xian et al. (US 6,327,584).

Claims 8, 9, 14, and 16 are rejected under 35 U.S.C. § 103(a) as unpatentable over Olsen (US 2001/0032300) in view of Knothe et al. (U.S. Pat. No. 4,811,293), and further in view of Hill et al. (US 2003/0221083).

Claim 20 is rejected under 35 U.S.C. § 103(a) as unpatentable over Olsen (US 2001/0032300) in view of Raves et al. (U.S. 2003/0182500).

Claims 8 and 12 stand rejected under 35 U.S.C. § 103(a) as being unpatentable Naomachi in view of Kato (U.S. Pat. No. 6,130,547).

The Examiner's rejections of the claims are respectfully traversed.

Response to Rejections of Claims Under 35 U.S.C. § 102/103

a. Claims 1-9

Claim 1 is rejected under 35 U.S.C. § 102 but cites a combination of references, which would be improper. The Applicant assumes the Examiner meant the rejection to be under 35 U.S.C. § 103(a), and the following remarks reflect this assumption.

Applicant's amended claim 1 recites:

A method of protecting memory locations associated with an embedded system, the method comprising:

starting a write filter that intercepts writes to the protected memory locations and stores the writes in a cache;

starting a state machine with at least a change state and a normal state;

upon starting the state machine, entering the change state when an indication is present that data needs to be persisted to the protected memory locations, otherwise entering the normal state;

in the normal state identifying requests for critical writes to the protected memory locations and creating at least one update file describing the critical writes, wherein the critical writes are not persisted to the protected memory locations during the normal state; and

in the change state, applying the critical writes described in the updated file and rebooting the system in a manner that persists the critical writes to the protected memory locations.

The Olsen Reference

The Examiner cites Olsen as disclosing "starting a write filter that intercepts writes to the protected memory locations and stores the writes in a cache" at Par. 23 Lines 11-16 and Par. 28 Lines 1-10, "starting a state machine with at least a change state and a normal state" at Par. 35 and Fig. 1c, "upon starting the state machine, entering the change state when an indication is present that data needs to be persisted to the protected memory locations, otherwise entering the normal state" at Par. 35, Fig. 1c Steps 22, 24 and 26, "in the normal state identifying requests for critical writes to the protected memory locations and creating at least one update file describing the critical writes" in Par 33 and 28, "in the change state, applying the critical writes described in the updated file and rebooting the system in a manner that persists the critical writes to the protected memory locations" in Par. 35 Fig. 1c steps 26 & 28.

Olsen does not teach or suggest "in the normal state identifying requests for critical writes to the protected memory locations and creating at least one update file describing the critical writes, wherein the critical writes are not persisted to the protected memory locations during the normal state." During the normal state, a lookaside buffer in the persistent volatile memory 36 is used to keep track of write transactions. As stated in Olsen Par. 34, when no write requests are pending, the lookaside buffer has a state of "0". When a write

request is received by the intermediary program 28, the intermediary program 28 stores the computed location and length in bytes of the request in the look-aside buffer and the state of the buffer becomes "1" (Step 12). Then, the actual contents of the write request are copied into the look-aside buffer (Step 14). If the copy is successfully completed, the look-aside buffer state becomes "2" (Step 16). Once the buffer state is set to "2", the contents of the write request are coped out of the look-aside buffer to their computed location in the persistent volatile memory 36 (Step 18). When this is successfully completed, the buffer state returns to "0" (Step 10). Thus, in the normal state, the critical writes are *always* persisted to their proper location in the persistent volatile memory 38 unless there is a system crash. Accordingly, Olsen does not teach or suggest the limitation "wherein the critical writes are not persisted to the protected memory locations during the normal state".

Olsen does not teach or suggest "in the change state, applying the critical writes described in the updated file and rebooting the system in a manner that persists the critical writes to the protected memory locations". In Olsen Par. 35, Lines 20-31 it states that if, during a boot of the computer 4, a state of the lookaside buffer is "2" (i.e., write data was fully transferred into the buffer but not yet fully transferred from the lookaside buffer into its location in the persistent volatile memory 36), "the intermediary program 28 copies the contents of the look-aside buffer to the computed location in the persistent volatile memory 36 (step 28). When the copy is completed, the buffer state is set to 0 (Step 20) and the intermediary program 28 checks the next look-aside buffer (Step 22). Eventually the intermediary program 28 will have checked the state of all the look-aside buffers, and the system boot will continue (Step 24)." Thus, Olsen does not teach or suggest "applying the critical writes described in the updated file *and rebooting the system*". Olsen *continues* with the current boot and does not include any step of "*rebooting* the system" as recited in Applicant's Claim 1. Accordingly, Olsen does not meet this limitation.

The Knothe et al. Reference

Knothe et al. does not make up for the deficiencies of Olsen in meeting

Applicant's Claim 1. Knothe et al. discloses a method in which any data desired can be inputted from a microprocessor into an electrically erasable memory during an initialization phase. The microprocessor defines at least a subarea of the electrically erasable memory as write-protected at the end of the initialization phase by setting at least one write protect flip-flop, the write protect flip-flop blocks the write line to the associated subarea of the electrically erasable memory in its logical state "with write protection" and the write protect flip-flop can only be brought into the logical stage "with write protection" by the microprocessor, while it can only be brought into the other logical state "without write protection" by the actuation of a mechanical switch element."

Knothe et al. does not teach or suggest "in the normal state identifying requests for critical writes to the protected memory locations and creating at least one update file describing the critical writes, wherein the critical writes are not persisted to the protected memory locations during the normal state." Knothe et al. does not provide any means for keeping track of changes to be written to the write-protected subarea of the electrically erasable memory after initialization. Knothe et al. simply blocks all writes to the write-protected subarea because no changes to the write-protected subarea are allowed after initialization. The purpose of the Knothe et al. system is to prevent access to the BIOS by applications after initialization. During initialization, parameters such as display type, graphics card, etc., may be changed (i.e., during the bootup of the computer) but after bootup these parameters are not allowed to be changed.

In contrast, Olsen's system stores write transactions in a look-aside table along with a state that indicates where the write transaction is in the process of writing the write data in the persistent volatile memory. Olsen's system would be made inoperable if altered to prevent all write transactions from being stored in the persistent volatile storage after initialization because 1) the look-aside table in Olsen is itself stored in the persistent volatile memory (i.e., the protected memory) and therefore if writes to the persistent volatile memory were blocked by Knothe et al.'s flip-flop mechanism after initialization, the table could not be updated; and 2) even if the table were stored elsewhere (i.e., not in the protected

memory), the writes would not be stored to the persistent volatile memory until a reboot occurred. Since in Olsen's system, reads are read only from their actual location in the persistent volatile memory without regard to the write transaction status from the look-aside table, subsequent read transactions would always read outdated data. Accordingly, Olsen even in combination with Knothe et al. does not meet the limitation "in the normal state identifying requests for critical writes to the protected memory locations and creating at least one update file describing the critical writes, wherein the critical writes are not persisted to the protected memory locations during the normal state" as recited in Applicant's Claim 1.

Knothe et al. also does not teach or suggest "in the change state, applying the critical writes described in the updated file and rebooting the system in a manner that persists the critical writes to the protected memory locations". In Knothe et al., there is no file that keeps track of critical writes and therefore no writes to persist in protected memory locations. In addition, Knothe et al. does not teach rebooting the system to persist (non-existing) critical writes after the initial boot. Accordingly, Olsen even in combination with Knothe et al. does not meet the limitation "in the change state, applying the critical writes described in the updated file and rebooting the system in a manner that persists the critical writes to the protected memory locations" as recited in Applicant's Claim 1.

Furthermore, none of Aviani, Xian, Hill or Raves meet the limitations missing from both Olsen and Knothe et al.

Accordingly, in view of the above, neither of Olsen or Knothe et al., nor any of the other references of record, taken either alone or in any combination, meets each and every limitation of Applicant's Claim 1, and therefore cannot be combined to formulate an obvious-type rejection under 35 U.S.C. § 103. Accordingly, Applicant respectfully submits that the 35 U.S.C. § 103 rejection of claim 1 should be withdrawn and that claim 1 is now in position for allowance.

Claims 2-9 each depend from independent base claim 1 and add further limitations. For at least the same reasons that Claim 1 is not shown, taught, or disclosed by the cited references, Claims 2-9 are likewise not shown, taught, or

disclosed. Thus, Applicant respectfully submits that the rejection of claims 2-9 should be withdrawn.

b. Claims 10-20

Claim 11 recites similar limitations to claim 1, including "during operation in the normal state, the applications are run and when a critical write to the operating system is requested, *the critical write is not persisted to the operating system but an update file is generated to store the critical write until the embedded system enters the change state*". For at least the same reasons that Claim 1 is not shown, taught, or disclosed by the cited references, Claim 10 is likewise not shown, taught, or disclosed. Thus, Applicant respectfully submits that the rejection of Claim 10 should be withdrawn.

c. Claims 11-20

Claim 11 recites similar limitations to claim 1, including "An embedded system that *in conjunction with booting* assumes one of two states: a normal state in which applications are executed and a write filter intercepts writes to a protected memory location and redirects them to a non-protected memory location *wherein the writes to the protected memory location are not applied to the protected memory location during the normal state, in which respective writes applied to the write filter during the last normal state are re-applied to the write filter and subsequently persisted to the respective protected memory locations*". For at least the same reasons that Claim 1 is not shown, taught, or disclosed by the cited references, Claim 11 is likewise not shown, taught, or disclosed. Thus, Applicant respectfully submits that the rejection of Claim 11 should be withdrawn.

Claims 12-20 each depend from independent base claim 11 and add further limitations. For at least the same reasons that Claim 11 is not shown, taught, or disclosed by the cited references, Claims 12-20 are likewise not shown, taught, or disclosed. Thus, Applicant respectfully submits that the rejection of claims 12-20 should be withdrawn.

Conclusion

In view of the foregoing remarks, it is respectfully submitted that none of the references cited by the Examiner taken alone or in any combination shows, teaches, or discloses the claimed invention, and that Claims 1-16 and 18-20 are in condition for allowance. Reexamination and reconsideration are respectfully requested.

Should the Examiner have any questions regarding this amendment, or should the Examiner believe that it would further prosecution of this application, the Examiner is invited to call the undersigned.

Respectfully submitted,

July 21, 2006

Jessica Costa

Jessica Costa, Reg. No. 41,065

The Law Offices of Jessica Costa, PC
P.O. Box 460 Crozet, VA
22932-0460
(434) 823-2232
(434) 823-2242 (fax)